PORTAL
US Patent & Trademark Office

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: ◉ The ACM Digital Library ○ The Guide

" snapshot garbage collection" and "advance marking" <near> SEARCH

THE ACM DIGITAL LIBRARY

Feedback Report a problem Satisfaction survey

09/993,382

Terms used **snapshot garbage collection** and **advance marking near stack sentence barrier** and **snapshot garbage collection**

Found **722** of **139,567**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

☐ Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Best 200 shown

Relevance scale ☐☐☐☐☐

1 Fast detection of communication patterns in distributed executions

Thomas Kunz, Michiel F. H. Seuren

November 1997 **Proceedings of the 1997 conference of the Centre for Advanced Studies on Collaborative research**

Full text available: pdf(4.21 MB) Additional Information: full citation, abstract, references, index terms

Understanding distributed applications is a tedious and difficult task. Visualizations based on process-time diagrams are often used to obtain a better understanding of the execution of the application. The visualization tool we use is Poet, an event tracer developed at the University of Waterloo. However, these diagrams are often very complex and do not provide the user with the desired overview of the application. In our experience, such tools display repeated occurrences of non-trivial commun ...

2 Breaking the complexity barrier again

Terry Winograd

November 1973 **Proceedings of the 1973 meeting on Programming languages and information retrieval**, Volume 9 , 10 Issue 3 , 1

Full text available: pdf(1.75 MB) Additional Information: full citation, references, citings

3 Illustrative risks to the public in the use of computer systems and related technology

Peter G. Neumann

January 1996 **ACM SIGSOFT Software Engineering Notes**, Volume 21 Issue 1

Full text available: pdf(2.54 MB) Additional Information: full citation

4 Discourse & pragmatics: Extending DRT with a focusing mechanism for pronominal anaphora and ellipsis resolution

José Abraços, José Gabriel Lopes

August 1994 **Proceedings of the 15th conference on Computational linguistics - Volume 2**

Full text available: pdf(409.40 KB) Additional Information: full citation, abstract, references

Cormack (1992) proposed a framework for pronominal anaphora resolution. Her proposal

integrates focusing theory (Sidner et al.) and DRT (Kamp and Reyle). We analyzed this methodology and adjusted it to the processing of Portuguese texts. The scope of the framework was widened to cover sentences containing restrictive relative clauses and subject ellipsis. Tests were conceived and applied to probe the adequacy of proposed modifications when dealing with processing of current texts.

5 Columns: Risks to the public in computers and related systems
Peter G. Neumann
January 2001 **ACM SIGSOFT Software Engineering Notes**, Volume 26 Issue 1
Full text available: pdf(3.24 MB) Additional Information: full citation

6 Human-computer interface development: concepts and systems for its management
H. Rex Hartson, Deborah Hix
March 1989 **ACM Computing Surveys (CSUR)**, Volume 21 Issue 1
Full text available: pdf(7.97 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Human-computer interface management, from a computer science viewpoint, focuses on the process of developing quality human-computer interfaces, including their representation, design, implementation, execution, evaluation, and maintenance. This survey presents important concepts of interface management: dialogue independence, structural modeling, representation, interactive tools, rapid prototyping, development methodologies, and control structures. *Dialogue independence* is th ...

7 A personal view of the personal work station: some firsts in the Fifties
Douglas Ross
January 1986 **Proceedings of the ACM Conference on The history of personal workstations**
Full text available: pdf(4.26 MB) Additional Information: full citation, references, index terms

8 IFIP Congress-62, Munich, Germany, August 27-September 1, 1962: Abstracts of papers
June 1962 **Communications of the ACM**, Volume 5 Issue 6
Full text available: pdf(1.35 MB) Additional Information: full citation

9 DOCUMENTS: an interactive online solution to four documentation problems
T. R. Girill, Clement H. Lulc
May 1983 **Communications of the ACM**, Volume 26 Issue 5
Full text available: pdf(1.14 MB) Additional Information: full citation, abstract, references, citings, index terms

An adequate delivery system for user documentation addresses the problems of easy access, versatile publication, convenient administration, and good document quality. At the National Magnetic Fusion Energy Computer Center the DOCUMENT program helps solve these problems by providing a high level of service through strategies that can readily be exported to other contexts. Dividing machine-readable documents into keyword windows permits fully online, subject-oriented ...

Keywords: help packages, information retrieval, keywords, online catalogs, user assistance, user interfaces

10 Interprocedural static analysis of sequencing constraints

Kurt M. Olender, Leon J. Osterweil

January 1992 **ACM Transactions on Software Engineering and Methodology (TOSEM)**, Volume 1 Issue 1

Full text available: pdf(2.37 MB) Additional Information: full citation, abstract, references, citings, index terms, review

This paper describes a system that automatically performs static interprocedural sequencing analysis from programmable constraint specifications. We describe the algorithms used for interprocedural analysis, relate the problems arising from the analysis of real-world programs, and show how these difficulties were overcome. Finally, we sketch the architecture of our prototype analysis system (called Cesar) and describe our experiences to date with its use, citing performance and error detect ...

Keywords: error detection, interprocedural data flow analysis, sequencing constraints

11 Computer Processing of Line-Drawing Images

Herbert Freeman

January 1974 **ACM Computing Surveys (CSUR)**, Volume 6 Issue 1

Full text available: pdf(3.18 MB) Additional Information: full citation, references, citings, index terms

12 A history of the Promis technology: an effective human interface

Jan Schultz

January 1986 **Proceedings of the ACM Conference on The history of personal workstations**

Full text available: pdf(2.61 MB) Additional Information: full citation, abstract, references, index terms

Scientific computing systems for individuals were pioneered early at Hewlett-Packard, beginning with the 9100A Desktop Calculator in 1968. Extensions of this first machine were soon seen in Personal Peripherals, such as Printers, Tape Cartridges, and Plotters, and followed by Graphic CRT Displays. By early 1972, the Desktop unit had been augmented by a very powerful Pocket Calculator, the ground-breaking HP 35A.This paper traces the evolution of these machines to the present day, ...

13 Tacit knowledge: icebergs in collaborative design

Brent N. Reeves, Frank Shipman

December 1996 **ACM SIGOIS Bulletin**, Volume 17 Issue 3

Full text available: pdf(1.28 MB) Additional Information: full citation, abstract, index terms

This workshop provides a forum for discussing experiences and issues related to tacit knowledge [Polanyi 1967] in collaborative systems. Beginning with early CSCW systems, tacit knowledge of work practice, in terms of unspoken assumptions and exceptions, has posed difficult problems for system designers. Analyses by Ehn [1988], Grudin [t994] and others [Bullen and Bennett 1990] show that tacit knowledge continues to play a disturbingly large role in the problems most CSCW systems struggle with. ...

14 An optimal algorithm for intersecting line segments in the plane

Bernard Chazelle, Herbert Edelsbrunner

January 1992 **Journal of the ACM (JACM)**, Volume 39 Issue 1

Full text available: pdf(3.84 MB) Additional Information: full citation, abstract, references, citings, index terms, review

The main contribution of this work is an O(n log n + k)-time algorithm for computing all k intersections among n line segments in the plane. This time complexity is easily shown to be optimal. Within the same asymptotic cost, our algorithm can also construct the subdivision of the plane defined by the segments and compute which segment (if any) lies right above (or below) each intersection ...

15 Pen computing: a technology overview and a vision

André Meyer

July 1995 **ACM SIGCHI Bulletin**, Volume 27 Issue 3

Full text available: pdf(5.14 MB) Additional Information: full citation, abstract, citings, index terms

This work gives an overview of a new technology that is attracting growing interest in public as well as in the computer industry itself. The visible difference from other technologies is in the use of a pen or pencil as the primary means of interaction between a user and a machine, picking up the familiar pen and paper interface metaphor. From this follows a set of consequences that will be analyzed and put into context with other emerging technologies and visions.Starting with a short historic ...

16 Testing the simplified decision model of telework: a quasi-experimental study

Stanley D. Clark, Lorne Olfman

April 2000 **Proceedings of the 2000 ACM SIGCPR conference on Computer personnel research**

Full text available: pdf(696.08 KB) Additional Information: full citation, abstract, references

Virtual organizations that serve customers via the Internet or online service networks have been growing at an incredible rate. These organizations do not require customers to come to the organization's physical location in order to do business. However, many more traditional organizations, even those that support virtual clientele, seem resistant to the use of virtual work arrangements. For organizations to become truly "virtual" they need to move to alternative work arrangemen ...

Keywords: decision model, simplified decision model, technology acceptance model, telework, virtual organization

17 Accesible interfaces: Using handhelds to help people with motor impairments

Brad A. Myers, Jacob O. Wobbrock, Sunny Yang, Brian Yeung, Jeffrey Nichols, Robert Miller

July 2002 **Proceedings of the fifth international ACM conference on Assistive technologies**

Full text available: pdf(1.03 MB) Additional Information: full citation, abstract, references, citings, index terms

People with Muscular Dystrophy (MD) and certain other muscular and nervous system disorders lose their gross motor control while retaining fine motor control. The result is that they lose the ability to move their wrists and arms, and therefore their ability to operate a mouse and keyboard. However, they can often still use their fingers to control a pencil or stylus, and thus can use a handheld computer such as a Palm. We have developed software that allows the handheld to substitute for the mo ...

Keywords: Muscular Dystrophy, Palm pilot, Pebbles, Personal Digital Assistants (PDAs), assistive technologies, disabilities, hand-held computers, handicapped

18 New article: Little machines: understanding users understanding interfaces

Johndan Johnson-Eilola

November 2001 **ACM Journal of Computer Documentation (JCD)**, Volume 25 Issue 4

Full text available: pdf(183.55 KB) Additional Information: full citation, abstract, references, index terms

This paper questions the ubiquitous practice of supplying minimalist information to users, of making that information functional only, of assuming that the Shannon-Weaver communication model should govern online systems, and of ignoring the social implications of such a stance. Help systems that provide fast, temporary solutions without providing any background information lead to the danger of users completing tasks that they do not understand at all. (Word will help us write a legal pleading, ...

Keywords: documentation, online help, user interface

19 Customer-focused design data in a large, multi-site organization

Paula Curtis, Tammy Heiserman, David Jobusch, Mark Notess, Jayson Webb

May 1999 **Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit**

Full text available: pdf(1.24 MB) Additional Information: full citation, references, citings, index terms

Keywords: affinity, contextual design, contextual inquiry, customer-focused design, distributed teams, organizational change, user data

20 Socially translucent systems: social proxies, persistent conversation, and the design of "babble"

Thomas Erickson, David N. Smith, Wendy A. Kellogg, Mark Laff, John T. Richards, Erin Bradner

May 1999 **Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit**

Full text available: pdf(1.07 MB) Additional Information: full citation, references, citings, index terms

Keywords: CMC, CSCW, IRC, awareness, chat, computer-medicated communication, conversation, design, discourse, social activity, social computing, visualization

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

The ACM Portal is published by the Association for Computing Machinery. Copyright ? 2004 ACM, Inc.
Terms of Usage Privacy Policy Code of Ethics Contact Us

Useful downloads: Adobe Acrobat QuickTime Windows Media Player Real Player

PORTAL
US Patent & Trademark Office

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: ◉ The ACM Digital Library ○ The Guide

garbage collection and (marking or flagging or indicating)<sen SEARCH

THE ACM DIGITAL LIBRARY

Feedback Report a problem Satisfaction survey

Terms used
garbage collection and **marking** or **flagging** or **indicating sentence stack** and **barrier**

Found **10,247** of **139,567**

Sort results by: relevance
Display results: expanded form

Save results to a Binder
Search Tips
☐ Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

09 | 993,382

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next
Best 200 shown

Relevance scale ☐☐☐☐☐

1 Garbage collection for a client-server persistent object store
Laurent Amsaleg, Michael J. Franklin, Olivier Gruber
August 1999 **ACM Transactions on Computer Systems (TOCS)**, Volume 17 Issue 3

Full text available: pdf(267.18 KB) Additional Information: full citation, abstract, references, citings, index terms, review

We describe an efficient server-based algorithm for garbage collecting persistent object stores in a client-server environmnet. The algorithm is incremental and runs concurrently with client transactions. Unlike previous algorithms, it does not hold any transactional locks on data and does non require callbacks to clients. It is fault-tolerant, but performs very little logging. The algorithm has been designed to be integrated into existing systems, and therefore it works with standard i ...

Keywords: client-server system, logging, persistent object-store, recovery

2 Collecting distributed garbage cycles by back tracing
Umesh Maheshwari, Barbara Liskov
August 1997 **Proceedings of the sixteenth annual ACM symposium on Principles of distributed computing**

Full text available: pdf(1.21 MB) Additional Information: full citation, references, citings, index terms

3 Garbage collecting the Internet: a survey of distributed garbage collection
Saleh E. Abdullahi, Graem A. Ringwood
September 1998 **ACM Computing Surveys (CSUR)**, Volume 30 Issue 3

Full text available: pdf(337.65 KB) Additional Information: full citation, abstract, references, citings, index terms, review

Internet programming languages such as Java present new challenges to garbage-collection design. The spectrum of garbage-collection schema for linked structures distributed over a network are reviewed here. Distributed garbage collectors are classified first because they evolved from single-address-space collectors. This taxonomy is used as a framework to explore distribution issues: locality of action, communication overhead and indeterministic communication latency.

Keywords: automatic storage reclamation, distributed, distributed file systems, distributed memories, distributed object-oriented management, memory management, network communication, object-oriented databases, reference counting

4 Garbage collection in a large LISP system



David A. Moon

August 1984 **Proceedings of the 1984 ACM Symposium on LISP and functional programming**

Full text available: pdf(1.25 MB) Additional Information: full citation, abstract, references, citings, index terms

This paper discusses garbage collection techniques used in a high-performance Lisp implementation with a large virtual memory, the Symbolics 3600. Particular attention is paid to practical issues and experience. In a large system problems of scale appear and the most straightforward garbage-collection techniques do not work well. Many of these problems involve the interaction of the garbage collector with demand-paged virtual memory. Some of the solutions adopted in the 3600 are presented, ...

5 Very concurrent mark-&-sweep garbage collection without fine-grain synchronization



Lorenz Huelsbergen, Phil Winterbottom

October 1998 **ACM SIGPLAN Notices , Proceedings of the first international symposium on Memory management**, Volume 34 Issue 3

Full text available: pdf(1.36 MB) Additional Information: full citation, abstract, references, citings, index terms

We describe a new incremental algorithm for the concurrent reclamation of a program's allocated, yet unreachable, data. Our algorithm is a variant of mark-&-sweep collection that---unlike prior designs---runs mutator, marker, and sweeper threads concurrently *without* explicit fine-grain synchronization on shared-memory multiprocessors. A global, but infrequent, synchronization coordinates the per-object coloring marks used by the three threads; fine-grain synchronization is achieve ...

6 Comparing mark-and sweep and stop-and-copy garbage collection



Benjamin Zorn

May 1990 **Proceedings of the 1990 ACM conference on LISP and functional programming**

Full text available: pdf(1.02 MB) Additional Information: full citation, abstract, references, citings, index terms

Stop-and-copy garbage collection has been preferred to mark-and-sweep collection in the last decade because its collection time is proportional to the size of reachable data and not to the memory size. This paper compares the CPU overhead and the memory requirements of the two collection algorithms extended with generations, and finds that mark-and-sweep collection requires at most a small amount of additional CPU overhead (3-6%) but, requires an average of 20% (and up to 40%) less memory t ...

7 An on-the-fly mark and sweep garbage collector based on sliding views



Hezi Azatchi, Yossi Levanoni, Harel Paz, Erez Petrank

October 2003 **ACM SIGPLAN Notices , Proceedings of the 18th ACM SIGPLAN conference on Object-oriented programing, systems, languages, and applications**, Volume 38 Issue 11

Full text available: pdf(244.12 KB) Additional Information: full citation, abstract, references, citings, index terms

With concurrent and garbage collected languages like Java and C# becoming popular, the need for a suitable non-intrusive, efficient, and concurrent multiprocessor garbage collector has become acute. We propose a novel mark and sweep on-the-fly algorithm based on the

sliding views mechanism of Levanoni and Petrank. We have implemented our collector on the Jikes Java Virtual Machine running on a Netfinity multiprocessor and compared it to the concurrent algorithm and to the stop-the-world collecto ...

Keywords: concurrent garbage collection, garbage collection, memory management, on-the-fly garbage collection, runtime systems

8 Concurrent garbage collection using hardware-assisted profiling

Timothy H. Heil, James E. Smith

October 2000 **ACM SIGPLAN Notices , Proceedings of the second international symposium on Memory management**, Volume 36 Issue 1

Full text available: pdf(1.74 MB) Additional Information: full citation, abstract, citings, index terms

In the presence of on-chip multithreading, a Virtual Machine (VM) implementation can readily take advantage of *service threads* for enhancing performance by performing tasks such as profile collection and analysis, dynamic optimization, and garbage collection concurrently with program execution. In this context, a hardware-assisted profiling mechanism is proposed. The *Relational Profiling Architecture* (RPA) is designed from the top down. RPA is based on a relational model similar ...

9 Older-first garbage collection in practice: evaluation in a Java Virtual Machine

Darko Stefanović, Matthew Hertz, Stephen M. Blackburn, Kathryn S. McKinley, J. Eliot B. Moss

June 2002 **ACM SIGPLAN Notices , Proceedings of the workshop on Memory system performance**, Volume 38 Issue 2 supplement

Full text available: pdf(1.15 MB) Additional Information: full citation, abstract, references, citings

Until recently, the best performing copying garbage collectors used a generational policy which repeatedly collects the very youngest objects, copies any survivors to an older space, and then infrequently collects the older space. A previous study that used garbage-collection simulation pointed to potential improvements by using an *Older-First* copying garbage collection algorithm. The Older-First algorithm sweeps a fixed-sized window through the heap from older to younger objects, and avo ...

10 Ulterior reference counting: fast garbage collection without a long wait

Stephen M. Blackburn, Kathryn S. McKinley

October 2003 **ACM SIGPLAN Notices , Proceedings of the 18th ACM SIGPLAN conference on Object-oriented programing, systems, languages, and applications**, Volume 38 Issue 11

Full text available: pdf(218.61 KB) Additional Information: full citation, abstract, references, index terms

General purpose garbage collectors have yet to combine short pause times with high throughput. For example, generational collectors can achieve high throughput. They have modest average pause times, but occasionally collect the whole heap and consequently incur long pauses. At the other extreme, concurrent collectors, including reference counting, attain short pause times but with significant performance penalties. This paper introduces a new hybrid collector that combines copying generational c ...

Keywords: Java, copying, generational hybrid, reference counting, ulterior reference counting

11 Systems: Myths and realities: the performance impact of garbage collection

Stephen M. Blackburn, Perry Cheng, Kathryn S. McKinley

June 2004 **Proceedings of the joint international conference on Measurement and modeling of computer systems**

Full text available: pdf(305.06 KB) Additional Information: full citation, abstract, references, index terms

This paper explores and quantifies garbage collection behavior for three whole heap collectors and generational counterparts: *copying semi-space, mark-sweep,* and *reference counting*, the canonical algorithms from which essentially all other collection algorithms are derived. Efficient implementations in MMTk, a Java memory management toolkit, in IBM's Jikes RVM share all common mechanisms to provide a clean experimental platform. Instrumentation separates collector and program behav ...

Keywords: generational, java, mark-sweep, reference counting, semi-space

12 Mostly concurrent garbage collection revisited

Katherine Barabash, Yoav Ossia, Erez Petrank

October 2003 **ACM SIGPLAN Notices , Proceedings of the 18th ACM SIGPLAN conference on Object-oriented programing, systems, languages, and applications**, Volume 38 Issue 11

Full text available: pdf(279.42 KB) Additional Information: full citation, abstract, references, index terms

The *mostly concurrent garbage collection* was presented in the seminal paper of Boehm et al. With the deployment of Java as a portable, secure and concurrent programming language, the mostly concurrent garbage collector turned out to be an excellent solution for Java's garbage collection task. The use of this collector is reported for several modern production Java Virtual Machines and it has been investigated further in academia.In this paper, we present a modification of the mostly concu ...

Keywords: JVM, Java, concurrent garbage collection, garbage collection, incremental garbage collection

13 Error-free garbage collection traces: how to cheat and not get caught

Matthew Hertz, Stephen M Blackburn, J Eliot B Moss, Kathryn S. McKinley, Darko Stefanović

June 2002 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the 2002 ACM SIGMETRICS international conference on Measurement and modeling of computer systems**, Volume 30 Issue 1

Full text available: pdf(105.06 KB) Additional Information: full citation, abstract, references, citings

Programmers are writing a large and rapidly growing number of programs in object-oriented languages such as Java that require garbage collection (GC). To explore the design and evaluation of GC algorithms quickly, researchers are using simulation based on traces of object allocation and lifetime behavior. The *brute force* method generates perfect traces using a whole-heap GC at every potential GC point in the program. Because this process is prohibitively expensive, researchers often use < ...

14 Beltway: getting around garbage collection gridlock

Stephen M Blackburn, Richard Jones, Kathryn S McKinley, J Eliot B Moss

May 2002 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2002 Conference on Programming language design and implementation**, Volume 37 Issue 5

Full text available: pdf(184.50 KB) Additional Information: full citation, abstract, references, citings, index terms

We present the design and implementation of a new garbage collection framework that significantly generalizes existing copying collectors. The *Beltway* framework exploits and separates object age and incrementality. It groups objects in one or more increments on queues called *belts*, collects belts independently, and collects increments on a belt in first-in-first-out order. We show that Beltway configurations, selected by command line options, act and perform the same as semi-space, ...

Keywords: Java, beltway, copying collection, generational collection

15 An experimental study of renewal-older-first garbage collection

Lars T. Hansen, William D. Clinger

September 2002 **ACM SIGPLAN Notices , Proceedings of the seventh ACM SIGPLAN international conference on Functional programming**, Volume 37 Issue 9

Full text available: pdf(143.87 KB) Additional Information: full citation, abstract, references, index terms

Generational collection has improved the efficiency of garbage collection in fast-allocating programs by focusing on collecting young garbage, but has done little to reduce the cost of collecting a heap containing large amounts of older data. A new generational technique, older-first collection, shows promise in its ability to manage older data.This paper reports on an implementation study that compared two older-first collectors to traditional (younger-first) generational collectors. One of the ...

Keywords: generational garbage collection, older-first

16 Cycles to recycle: garbage collection to the IA-64

Richard L. Hudson, J. Elliot Moss, Sreenivas Subramoney, Weldon Washburn

October 2000 **ACM SIGPLAN Notices , Proceedings of the second international symposium on Memory management**, Volume 36 Issue 1

Full text available: pdf(1.25 MB) Additional Information: full citation, abstract, citings, index terms

The IA-64, Intel's 64-bit instruction set architecture, exhibits a number of interesting architectural features. Here we consider those features as they relate to supporting garbage collection (GC). We aim to assist GC and compiler implementors by describing how one may exploit features of the IA-64. Along the way, we record some previously unpublished object scanning techniques, and offer novel ones for object allocation (suggesting some simple operating system support that would simplify it ...

17 Interactive Editing Systems: Part II

Norman Meyrowitz, Andries van Dam

September 1982 **ACM Computing Surveys (CSUR)**, Volume 14 Issue 3

Full text available: pdf(9.17 MB) Additional Information: full citation, references, citings, index terms

18 Age-based garbage collection

Darko Stefanović, Kathryn S. McKinley, J. Eliot B. Moss

October 1999 **ACM SIGPLAN Notices , Proceedings of the 14th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications**, Volume 34 Issue 10

Full text available: pdf(1.47 MB) Additional Information: full citation, abstract, references, citings, index terms

Modern generational garbage collectors look for garbage among the young objects, because they have high mortality; however, these objects include the very youngest objects, which clearly are still live. We introduce new garbage collection algorithms, called age-based, some of which postpone consideration of the youngest objects. Collecting less than the whole heap requires write barrier mechanisms to track pointers into the collected region. We describe her ...

Keywords: garbage collection, generational and copy collection, object behavior, write barrier

19 Implementing an on-the-fly garbage collector for Java

Tamar Domani, Elliot K. Kolodner, Ethan Lewis, Eliot E. Salant, Katherine Barabash, Itai Lahan, Yossi Levanoni, Erez Petrank, Igor Yanorer

October 2000 **ACM SIGPLAN Notices , Proceedings of the second international symposium on Memory management**, Volume 36 Issue 1

Full text available: pdf(1.33 MB) Additional Information: full citation, abstract, citings, index terms

Java uses garbage collection (GC) for the automatic reclamation of computer memory no longer required by a running application. GC implementations for Java Virtual Machines (JVM) are typically designed for single processor machines, and do not necessarily perform well for a server program with many threads running on a multiprocessor. We designed and implemented an on-the-fly GC, based on the algorithm of Doligez, Leroy and Gonthier [13, 12] (DLG), for Java in this environment. An *on-the-f ...*

Keywords: *Java, concurrent garbage collection, garbage collection, memory management, on-the-fly garbage collection, programming languages*

20 Session 3: languages: Efficient parallel global garbage collection on massively parallel computers

Tomio Kamada, Satoshi Matsuoka, Akinori Yonezawa

November 1994 **Proceedings of the 1994 ACM/IEEE conference on Supercomputing**

Full text available: pdf(1.06 MB) Additional Information: full citation, abstract, references

On distributed-memory high-performance MPPs where processors are interconnected by an asynchronous network, efficient Garbage Collection (GC) becomes difficult due to inter-node references and references within pending, unprocessed messages. Our parallel global GC algorithm (1) takes advantage of reference locality, (2) efficiently traverses references over nodes, (3) admits minimum pause time of ongoing computations, and (4) has been shown to scale up to 1024 node MPPs. The algorithm employs a ...

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

The ACM Portal is published by the Association for Computing Machinery. Copyright ? 2004 ACM, Inc.
Terms of Usage Privacy Policy Code of Ethics Contact Us

Useful downloads: Adobe Acrobat QuickTime Windows Media Player Real Player

PORTAL
US Patent & Trademark Office

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: The ACM Digital Library The Guide

garbage collection and (marking or flagging or indicating) <nea SEARCH

THE ACM DIGITAL LIBRARY

Feedback Report a problem Satisfaction survey

Terms used
garbage collection and **marking** or **flagging** or **indicating near stack** and **barrier** and **snapshot garbage collection**

Found **11,453** of **139,567**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Best 200 shown

Relevance scale

1 Garbage collection for a client-server persistent object store

Laurent Amsaleg, Michael J. Franklin, Olivier Gruber

August 1999 **ACM Transactions on Computer Systems (TOCS)**, Volume 17 Issue 3

Full text available: pdf(267.18 KB) Additional Information: full citation, abstract, references, citings, index terms, review

We describe an efficient server-based algorithm for garbage collecting persistent object stores in a client-server environmnet. The algorithm is incremental and runs concurrently with client transactions. Unlike previous algorithms, it does not hold any transactional locks on data and does non require callbacks to clients. It is fault-tolerant, but performs very little logging. The algorithm has been designed to be integrated into existing systems, and therefore it works with standard i ...

Keywords: client-server system, logging, persistent object-store, recovery

2 Cycles to recycle: garbage collection to the IA-64

Richard L. Hudson, J. Elliot Moss, Sreenivas Subramoney, Weldon Washburn

October 2000 **ACM SIGPLAN Notices , Proceedings of the second international symposium on Memory management**, Volume 36 Issue 1

Full text available: pdf(1.25 MB) Additional Information: full citation, abstract, citings, index terms

The IA-64, Intel's 64-bit instruction set architecture, exhibits a number of interesting architectural features. Here we consider those features as they relate to supporting garbage collection (GC). We aim to assist GC and compiler implementors by describing how one may exploit features of the IA-64. Along the way, we record some previously unpublished object scanning techniques, and offer novel ones for object allocation (suggesting some simple operating system support that would simplify it ...

3 Garbage collecting the Internet: a survey of distributed garbage collection

Saleh E. Abdullahi, Graem A. Ringwood

September 1998 **ACM Computing Surveys (CSUR)**, Volume 30 Issue 3

Full text available: pdf(337.65 KB) Additional Information: full citation, abstract, references, citings, index terms, review

Internet programming languages such as Java present new challenges to garbage-collection design. The spectrum of garbage-collection schema for linked structures distributed over a

network are reviewed here. Distributed garbage collectors are classified first because they evolved from single-address-space collectors. This taxonomy is used as a framework to explore distribution issues: locality of action, communication overhead and indeterministic communication latency.

Keywords: automatic storage reclamation, distributed, distributed file systems, distributed memories, distributed object-oriented management, memory management, network communication, object-oriented databases, reference counting

4 A comparative performance evaluation of write barrier implementation

Antony L. Hosking, J. Eliot B. Moss, Darko Stefanovic

October 1992 **ACM SIGPLAN Notices , conference proceedings on Object-oriented programming systems, languages, and applications**, Volume 27 Issue 10

Full text available: pdf(2.48 MB) Additional Information: full citation, references, citings, index terms

5 Collecting distributed garbage cycles by back tracing

Umesh Maheshwari, Barbara Liskov

August 1997 **Proceedings of the sixteenth annual ACM symposium on Principles of distributed computing**

Full text available: pdf(1.21 MB) Additional Information: full citation, references, citings, index terms

6 Controlling fragmentation and space consumption in the metronome, a real-time garbage collector for Java

David F. Bacon, Perry Cheng, V. T. Rajan

June 2003 **ACM SIGPLAN Notices , Proceedings of the 2003 ACM SIGPLAN conference on Language, compiler, and tool for embedded systems**, Volume 38 Issue 7

Full text available: pdf(354.15 KB) Additional Information: full citation, abstract, references, index terms

Now that the use of garbage collection in languages like Java is becoming widely accepted due to the safety and software engineering benefits it provides, there is significant interest in applying garbage collection to hard real-time systems. Past approaches have generally suffered from one of two major flaws: either they were not provably real-time, or they imposed large space overheads to meet the real-time bounds.Our previous work [3] presented the Metronome, a mostly non-copying real-time co ...

Keywords: compaction, cost model, fragmentation, space bounds

7 Fast floating-point processing in Common Lisp

Richard J. Fateman, Kevin A. Broughan, Diane K. Willcock, Duane Rettig

March 1995 **ACM Transactions on Mathematical Software (TOMS)**, Volume 21 Issue 1

Full text available: pdf(2.58 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Lisp, one of the oldest higher-level programming languages, has rarely been used for fast numerical (floating-point) computation. We explore the benefits of Common Lisp, an emerging new language standard with some excellent implementations, for numerical computation. We compare it to Fortran in terms of the speed of efficiency of generated code, as well as the structure and convenience of the language. There are a surprising number of advantages to Lisp, especially in cases where a mixture ...

Keywords: C programming language, Common Lisp, Fortran, Lisp, compiler optimization,

floating-point arithmetic, numerical algorithms, symbolic computation

8 Beltway: getting around garbage collection gridlock

Stephen M Blackburn, Richard Jones, Kathryn S McKinley, J Eliot B Moss

May 2002 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2002 Conference on Programming language design and implementation**, Volume 37 Issue 5

Full text available: pdf(184.50 KB) Additional Information: full citation, abstract, references, citings, index terms

We present the design and implementation of a new garbage collection framework that significantly generalizes existing copying collectors. The *Beltway* framework exploits and separates object age and incrementality. It groups objects in one or more increments on queues called *belts*, collects belts independently, and collects increments on a belt in first-in-first-out order. We show that Beltway configurations, selected by command line options, act and perform the same as semi-space, ...

Keywords: Java, beltway, copying collection, generational collection

9 A parallel, incremental and concurrent GC for servers

Yoav Ossia, Ori Ben-Yitzhak, Irit Goft, Elliot K. Kolodner, Victor Leikehman, Avi Owshanko

May 2002 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2002 Conference on Programming language design and implementation**, Volume 37 Issue 5

Full text available: pdf(231.80 KB) Additional Information: full citation, abstract, references, citings, index terms

Multithreaded applications with multi-gigabyte heaps running on modern servers provide new challenges for garbage collection (GC). The challenges for "server-oriented" GC include: ensuring short pause times on a multi-gigabyte heap, while minimizing throughput penalty, good scaling on multiprocessor hardware, and keeping the number of expensive multi-cycle fence instructions required by weak ordering to a minimum. We designed and implemented a fully parallel, incremental, mostly concurrent colle ...

Keywords: JVM, Java, concurrent garbage collection, garbage collection, incremental garbage collection, weak ordering

10 A compacting incremental collector and its performance in a production quality compiler

Martin Larose, Marc Feeley

October 1998 **ACM SIGPLAN Notices , Proceedings of the first international symposium on Memory management**, Volume 34 Issue 3

Full text available: pdf(1.20 MB) Additional Information: full citation, abstract, references, citings, index terms

We present a new near-real-time compacting collector and its implementation in a production quality Scheme compiler (Gambit-C). Our goal is to use this system as a base for an implementation of Erlang for writing soft real-time telecommunication applications. We start with a description of Gambit-C's memory organisation and its blocking collector. The design and integration of the incremental collector within Gambit-C are then explained. Finally we measure the performance of the incremental coll ...

11 Heap compression for memory-constrained Java environments

G. Chen, M. Kandemir, N. Vijaykrishnan, M. J. Irwin, B. Mathiske, M. Wolczko

October 2003 **ACM SIGPLAN Notices , Proceedings of the 18th ACM SIGPLAN conference on Object-oriented programing, systems, languages, and**

applications, Volume 38 Issue 11

Full text available: pdf(2.14 MB) Additional Information: full citation, abstract, references, citings, index terms

Java is becoming the main software platform for consumer and embedded devices such as mobile phones, PDAs, TV set-top boxes, and in-vehicle systems. Since many of these systems are memory constrained, it is extremely important to keep the memory footprint of Java applications under control.The goal of this work is to enable the execution of Java applications using a smaller heap footprint than that possible using current embedded JVMs. We propose a set of memory management strategies to reduce h ...

Keywords: Java virtual machine, garbage collection, heap, memory compression

12 Understanding the connectivity of heap objects

Martin Hirzel, Johannes Henkel, Amer Diwan, Michael Hind

June 2002 **ACM SIGPLAN Notices , Proceedings of the third international symposium on Memory management**, Volume 38 Issue 2 supplement

Full text available: pdf(256.15 KB) Additional Information: full citation, abstract, references, citings, index terms

Modern garbage collectors partition the set of heap objects to achieve the best performance. For example, generational garbage collectors partition objects by age and focus their efforts on the youngest objects. Partitioning by age works well for many programs because younger objects usually have short lifetimes and thus garbage collection of young objects is often able to free up many objects. However, generational garbage collectors are typically much less efficient for longer-lived objects, a ...

Keywords: connectivity based garbage collection, object lifetimes

13 Creating and preserving locality of java applications at allocation and garbage collection times

Yefim Shuf, Manish Gupta, Hubertus Franke, Andrew Appel, Jaswinder Pal Singh

November 2002 **ACM SIGPLAN Notices , Proceedings of the 17th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications**, Volume 37 Issue 11

Full text available: pdf(180.20 KB) Additional Information: full citation, abstract, references, citings, index terms

The growing gap between processor and memory speeds is motivating the need for optimization strategies that improve data locality. A major challenge is to devise techniques suitable for pointer-intensive applications. This paper presents two techniques aimed at improving the memory behavior of pointer-intensive applications with dynamic memory allocation, such as those written in Java. First, we present an allocation time object placement technique based on the recently introduced notion of *p* ...

Keywords: *JVM, Java, garbage collection, heap traversal, locality, locality based graph traversal, memory allocation, memory management, object co-allocation, object placement, prolific types, run-time systems*

14 Support for garbage collection at every instruction in a Java compiler

James M. Stichnoth, Guei-Yuan Lueh, Michał Cierniak

May 1999 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1999 conference on Programming language design and implementation**, Volume 34 Issue 5

Full text available: pdf(1.06 MB) Additional Information: full citation, abstract, references, citings, index

terms

A high-performance implementation of a Java Virtual Machine[1] requires a compiler to translate Java bytecodes into native instructions, as well as an advanced garbage collector (e.g., copying or generational). When the Java heap is exhausted and the garbage collector executes, the compiler must report to the garbage collector all live object references contained in physical registers and stack locations. Typical compilers only allow certain instructions (e.g., call instructions and bac ...

Keywords: Java, compilers, garbage collection

15 Garbage collection in a large LISP system

David A. Moon

August 1984 **Proceedings of the 1984 ACM Symposium on LISP and functional programming**

Full text available: pdf(1.25 MB) Additional Information: full citation, abstract, references, citings, index terms

This paper discusses garbage collection techniques used in a high-performance Lisp implementation with a large virtual memory, the Symbolics 3600. Particular attention is paid to practical issues and experience. In a large system problems of scale appear and the most straightforward garbage-collection techniques do not work well. Many of these problems involve the interaction of the garbage collector with demand-paged virtual memory. Some of the solutions adopted in the 3600 are presented, ...

16 Safely executing untrusted code: Upgrading transport protocols using untrusted mobile code

Parveen Patel, Andrew Whitaker, David Wetherall, Jay Lepreau, Tim Stack

October 2003 **Proceedings of the nineteenth ACM symposium on Operating systems principles**

Full text available: pdf(248.86 KB) Additional Information: full citation, abstract, references, index terms

In this paper, we present STP, a system in which communicating end hosts use untrusted mobile code to remotely upgrade each other with the transport protocols that they use to communicate. New transport protocols are written in a type-safe version of C, distributed out-of-band, and run in-kernel. Communicating peers select a transport protocol to use as part of a TCP-like connection setup handshake that is backwards-compatible with TCP and incurs minimum connection setup latency. New transports ...

Keywords: TCP-friendliness, deployment, implementation, transport protocols, untrusted mobile code

17 Eager Haskell: resource-bounded execution yields efficient iteration

Jan-Willem Maessen

October 2002 **Proceedings of the ACM SIGPLAN workshop on Haskell**

Full text available: pdf(161.87 KB) Additional Information: full citation, abstract, references, citings, index terms

The advantages of the Haskell programming language are rooted in its clean equational semantics. Those advantages evaporate as soon as programmers try to write simple iterative computations and discover that their code must be annotated with calls to *seq* in order to overcome space leaks introduced by lazy evaluation. The Eager Haskell compiler executes Haskell programs eagerly by default, *i.e.*, bindings and function arguments are evaluated before bodies. When resource bounds are ex ...

18 Design of the Mneme persistent object store

J. Eliot B. Moss

April 1990 **ACM Transactions on Information Systems (TOIS)**, Volume 8 Issue 2

Full text available: pdf(3.22 MB) Additional Information: full citation, abstract, references, citings, index terms, review

The Mneme project is an investigation of techniques for integrating programming language and database features to provide better support for cooperative, information-intensive tasks such as computer-aided software engineering. The project strategy is to implement efficient, distributed, persistent programming languages. We report here on the Mneme persistent object store, a fundamental component of the project, discussing its design and initial prototype. Mneme stores objects

19 Very concurrent mark-&-sweep garbage collection without fine-grain synchronization

Lorenz Huelsbergen, Phil Winterbottom

October 1998 **ACM SIGPLAN Notices , Proceedings of the first international symposium on Memory management**, Volume 34 Issue 3

Full text available: pdf(1.36 MB) Additional Information: full citation, abstract, references, citings, index terms

We describe a new incremental algorithm for the concurrent reclamation of a program's allocated, yet unreachable, data. Our algorithm is a variant of mark-&-sweep collection that---unlike prior designs---runs mutator, marker, and sweeper threads concurrently *without* explicit fine-grain synchronization on shared-memory multiprocessors. A global, but infrequent, synchronization coordinates the per-object coloring marks used by the three threads; fine-grain synchronization is achieve ...

20 Ulterior reference counting: fast garbage collection without a long wait

Stephen M. Blackburn, Kathryn S. McKinley

October 2003 **ACM SIGPLAN Notices , Proceedings of the 18th ACM SIGPLAN conference on Object-oriented programing, systems, languages, and applications**, Volume 38 Issue 11

Full text available: pdf(218.61 KB) Additional Information: full citation, abstract, references, index terms

General purpose garbage collectors have yet to combine short pause times with high throughput. For example, generational collectors can achieve high throughput. They have modest average pause times, but occasionally collect the whole heap and consequently incur long pauses. At the other extreme, concurrent collectors, including reference counting, attain short pause times but with significant performance penalties. This paper introduces a new hybrid collector that combines copying generational c ...

Keywords: Java, copying, generational hybrid, reference counting, ulterior reference counting

The ACM Portal is published by the Association for Computing Machinery. Copyright ? 2004 ACM, Inc.
Terms of Usage Privacy Policy Code of Ethics Contact Us

Useful downloads: Adobe Acrobat QuickTime Windows Media Player Real Player